Adapting Control Methods for Autonomous Exploration of Unknown Environments

Barbara Engelhardt, Steve Chien, Darren Mutz
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 91109-8099
{firstname.lastname}@jpl.nasa.gov

Abstract

Proposed missions to explore comets and moons will encounter environments that are hostile and unpredictable. Any successful explorer must be able to adapt to a wide range of possible operating conditions in order to survive. The traditional approach of constructing special-purpose control methods would require information about the environment, which is not available a priori for these missions. An alternate approach is to utilize general control with significant capability to adapt its behavior, a so called *adaptive problem-solving* methodology. Using adaptive problem-solving, a spacecraft can use reinforcement learning to adapt an environment-specific search strategy given the craft's general problem solver with a flexible control architecture.

Introduction

Because of light-time communication delays, exploration missions require an autonomous explorer that can adapt to handle possible environments. For autonomous planning systems, the high-level actions of the spacecraft must be planned with sufficient environmental information to ensure that the resulting plans are admissible. A spacecraft could easily be lost based on inappropriate behavior in a particular environment due to overly-generic control methods (Minton 1996).

On the other hand, developing and testing domain-specific control methods is extremely difficult, and requires support of a domain expert. Moreover, the domain expert must have knowledge about the environment in which the spacecraft is operating, which is not available before the spacecraft arrives at the location to explore. If experts are not available, the spacecraft must be able to automatically adapt a flexible control structure specific to the new environment.

Adaptive problem solving addresses these problems by enabling the development and maintenance of effective control strategies without extensive domain-specific knowledge. An adaptive problem solver is given: (1) a generic set of control strategies and (2) a flexible control architecture, and uses a statistical method to estimate the quality of each control strategy or generate a more appropriate strategy. Adaptive problem solving also provides hard statistical guarantees on the quality of the behavior for each adapted control method. Using adaptive problem solving techniques, spacecraft exploration in unknown environments becomes feasible.

In this paper, we describe how adaptive problem solving can be used to adapt the control methods of a spacecraft in-situ. The value of this method is empirically shown in the context of two spacecraft operations scheduling problems in a generic planning and scheduling environment. By adapting control strategies for each domain, the lifespan of the spacecraft is improved since the adaptive problem solver can increase chances of spacecraft survival and continue to update the control methods based on aging hardware or environmental changes.

Motivational Example

The comet lander will land on a surface of unknown density, with the goals of drilling into the comet 90% and imaging its surroundings 10% of the time allocated to accomplishing goals. Situations will force these percentages to be innapropriate. One scenario might be that the surface of the comet is much denser than expected, so the rate of drilling is decreased and the wear on the drill is increased. The lander might need to adjust its priorities to take more images instead of drilling. Another scenario might be that drilling caused a layer of dust on the surface to drift up, the dust might limit the visibility of the lander. Taking images might be ineffective, so the lander might want to delay its drilling activities until the dust settles, or take images before drilling.

Failure to adapt to these situations could cost the lander the mission, by depleting resources too rapidly, not accomplishing mission objectives, or wearing out equipment. Not all possible situations can be enumerated before the mission; instead an adaptive problem solver checks the current control strategy's performance in the given environment and responds to changes by adapting the control strategy, independent of the cause of the change. An adaptive problem solver would continually adapt the control strategy if it found the current strategy non-optimal.

Planning System

The planning and scheduling system used to evaluate the control strategies for each model is a version of the ASPEN (Automated Scheduling and Planning ENvironment) system (Fukunaga *et al.* 1997). ASPEN is a configurable, generic planning/scheduling application framework that can be tailored to specific domains to create feasible schedules.

ASPEN employs planning and scheduling techniques to automatically generate a necessary activity sequence to achieve the mission goals. This sequence is produced by utilizing an iterative repair algorithm (Zweben *et al.* 1994) which classifies conflicts and attacks them individually. Conflicts occur when a plan constraint has been violated where this constraint could be temporal or involve a resource, state or activity parameter. Conflicts are resolved by performing schedule modifications such as moving, adding, or deleting activities. The target of the repair modification is chosen by a heuristic method, and the point in the search where this choice is made is called a choice point. For each type of choice point, the user creates a set of heuristic methods to use with varying usage weights. The set of heuristic methods impacts the outcome of the schedule, and effectively controls the behavior of the spacecraft.

The control strategies for adaptive problem solving are represented as sets of weighted heuristics so that they may be robust enough to perform well over the entire problem distribution even when they are slightly suboptimal, as opposed to a single heuristic which may not be as flexible to environment or hardware changes.

The quality of a resulting schedule generated by ASPEN is measured by a set of preferences specified by the user. This set of preferences specifies the quality functions associated with certain parameters in the schedule.

Adapting Control Strategies

To adapt control strategies, we can search the neighborhood of a current strategy, and select higher-scoring strategies. Given a set of possible control strategies, the adaptive problem solver selects the top strategies based on collecting samples of spacecraft performance in the current environment by running ASPEN evaluating the resulting schedule. The top strategies are returned to the search algorithm, which produces a subsequent set of hypotheses based on previously selected hypotheses using algorithm-specific techniques. This cycle continues until a certain amount of time has passed or another stopping criterion of the specific search algorithm has been met (see figure 1).

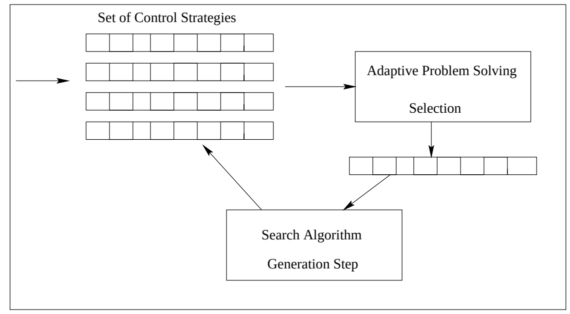

Figure 1: Hypothesis Generation Diagram

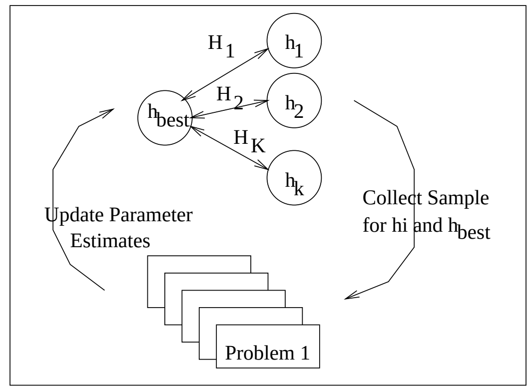

Figure 2: Adaptive Problem Solving Diagram

Adaptive Problem Solving

The adaptive problem solver attempts to select the top strategies from a set of strategies, supplied by the search algorithm, whose quality is a function of unknown parameters. It makes estimates of the parameters for strategy utility and sample cost in order to achieve a requested accuracy for a statistical decision requirement. The adaptive problem solver iteratively refines the utility and cost parameter estimates by acquiring training examples and reevaluating utility and cost (see figure 2).

The normal parametric model for reasoning about statistical error is used in this analysis, which assumes that the difference between the expected utility and estimated utility of a hypothesis can be accurately approximated by a normal distribution. This assumption is grounded in the Central Limit Theorem and is further discussed in (Chien *et al.* 1995). The analysis would change given a different parametric model, but the results should be analogous for conventional models.

Since parameter estimates are refined by random sampling, it is impossible to place perfect accuracy requirements on the selection algorithms. In practice, probabilistic requirements, or *decision criteria*, on the relative accuracy of the parameter estimates (and subsequent strategy selection) are chosen as parameterized forms that allow a tradeoff between accuracy and cost.

Specifically, decision requirements take a set of hypotheses and a probabilistic error bound, and terminate when one of the hypotheses can be shown to have the greatest mean, evaluated through pair-wise comparisons, with a confidence higher than the given confidence level. The overall confidence for selection is a function of the confidence of each pair-wise comparison. Rational analysis can be used to allocate error to each pairwise comparison in such a way as to attempt to optimize the resource usage necessary to acquire a sufficient number of samples across the comparisons to achieve the decision requirement (Gratch & DeJong 1994).

In this analysis, the decision requirement that is used in the adaptive problem solver is the probably approximately correct (PAC) requirement. The choice of using PAC in this analysis is mostly based on its prevalence rather than specific attributes of the requirement. The expected loss decision requirement was evaluated and found to have minimal impact on the outcome.

PAC Requirement

In order to satisfy the PAC requirement, the hypothesis estimated to be the best must be within some user-specified constant ϵ distance of the true best hypothesis with probability $1 - \delta$. The sum of the error from each pair-wise comparison is bounded by this probability. Let H_{sel} be the expected utility of the selected hypotheses and H_i be the expected utility for the remaining hypotheses. Let $\hat{H}$ be the estimate of the expected utility of a hypothesis. It is sufficient to bound the probability of error in selection for pair-wise comparisons with the following equation:

$$\sum_{i=1}^{k-1} Pr[\hat{H}_i < \hat{H}_{sel} - \epsilon | H_i > H_{sel} + \epsilon] \leq \delta \qquad (1)$$

Thus the problem of bounding the overall error reduces to bounding the error of each $k-1$ comparisons of the chosen best hypothesis to the rest of the hypotheses.

The normality assumption reduces equation 1 to a function of the parameter estimates, the number of examples n used to refine the estimates, the closeness parameter ϵ, and an unknown variance term σ^2. The two stopping criteria for selection are *dominance*, which is based on achieving a probability (δ) through sampling that h_i will perform better on a specific problem than h_j, and *indifference*, which is the probability that the difference between performances will fall within ϵ of 0. For the rest of this discussion, ϵ is ignored to simplify understanding. The equation for the probability of incorrect selection for a pair-wise comparison, α_i, is:

$$\alpha_i = \Phi\left(-(H_{sel} - H_i)\frac{\sqrt{n}}{\sqrt{\sigma^2_{sel,i}}}\right) \qquad (2)$$

We can use this relationship to determine how many training examples to allocate to each comparison, given the error bound on the probability of a mistake, an estimate of the difference in expected utility, and an estimate of the variance of each hypothesis:

$$n_{sel,i} = \frac{\sigma^2_{sel,i}}{(H_{sel} - H_i)^2}[\Phi^{-1}(\alpha_i)]^2 \qquad (3)$$

Rational Analysis

The hypothesis selection algorithm as presented does not take advantage of unequal distribution of error. By distributing error unequally across the pair-wise comparisons using the estimates of the sample cost and utility parameters, we can attempt to satisfy the requirements using the minimum possible cost. The general idea of rational analysis is to choose the error α_i for each comparison to minimize, subject to the given decision requirements:

$$\sum_{i=1}^{k-1} c_{sel,i} n_{sel,i}$$

The algorithm must only ensure that the *sum* of the errors remains less than the given bound. If one pair-wise comparison requires many more samples to achieve the same amount of accuracy as another comparison, then if the first is allowed to have more error and the second is allowed less, the overall cost of achieving those local requirements might be reduced. In practice, this method significantly reduces the number of samples necessary to achieve the requirement for certain domains, as shown in (Gratch & DeJong 1994).

Adapting Hypotheses

In order to adapt hypotheses, search algorithms are used to generate hypotheses in the neighborhood of the given hypotheses. At each level of search, an adaptive problem solving algorithm is used to evaluate the competing hypotheses with a given confidence bound.

Local Beam Search

One algorithm used to generate and search over hypotheses is local beam search (Russell & Norvig 1995). In a flexible planning and scheduling domain, each hypothesis, or combination of heuristics, can be represented as a vector of percentages where the percentages of heuristics associated with a certain type of choice point in ASPEN sum to 100% (see figure 3). A random heuristic is included for each plan problem. The basic algorithm is included below.

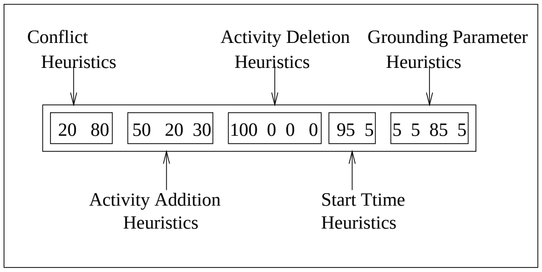

Figure 3: Hypothesis Vector Diagram

We chose a neighborhood of a vector to be defined as, for each subset of heuristics associated with a certain choice point, changing one of the usage percentages by a certain range, and scaling all of the other usage percentages equal amounts so that the sum is still 100% (see figure 3). Let l be the bound on the number of hypotheses the adaptive problem solver can evaluate.

Genetic Algorithm

Another algorithm that is used to generate hypotheses is a genetic algorithm (Goldberg 1989). Each hypothesis is represented as a vector of percentages, as in the local beam search. The three general operators (crossover, mutation, and reproduction) are used to generate the next set of hypotheses to search over, and ranking the hypotheses is done using adaptive problem solving. The crossover operator is not aware of the different subsets of heuristics, and may choose to split within one of those subsets. Mutation also works without knowledge of the constraint that subsets must sum to 100%, so each subset is scaled to 100 uniformly after the mutation operator is run. The basic algorithm is shown below.

Method Implementation

An adaptive control system of this type can be used in mission operations in multiple capacities. It can be used from the start to design spacecraft constraints and payload, by evaluating each of the potential designs against possible environments and comparing results. It can be used on the ground to perform mission planning and during flight to quickly develop new schedules based on changing domains or spacecraft deterioration. Environmental constraints for the spacecraft, such as the density or temperature of the surface for a lander, can be determined when they are available to the spacecraft. These constraints can be used to update the model of the environment, and adaptive problem solving can be used to efficiently determine the optimal planning heuristics for the current environment.

Empirical Evaluation

We claim that hypothesis adaptation can efficiently find a better set of hypotheses in a given domain. In this section we provide evidence that in practice, these methods can generate heuristic sets superior to those generated by model experts.

The test of real-world applicability is based on two domains related to planned space missions, using the ASPEN planning and scheduling system. The original set of hypotheses that is used is the set of heuristic combinations currently in use in these and related models. We hope this illustrates how this type of method can be useful in real-world domains, by improving on control strategies already in use or updating the strategies to handle domain shifts.

Evaluation

New Millennium EO-1 Domain – New Millennium Earth Observer 1 (EO-1) is an earth imaging satellite featuring an advanced multi-spectral imaging device. EO-1 mission operations consists of managing spacecraft operability constraints (power, thermal, pointing, buffers, consumables, telecommunications, etc.) and science goals (imaging of specific targets within particular observation parameters). The EO-1 domain models the operations of the EO-1 operations for a two-day horizon (Sherwood *et al.* 1998). It consists of 14 resources, 10 state variables and 38 different activity types. Each EO-1 problem instance includes a randomly generated, fixed profile that represents typical weather and instrument pattern. Each problem also includes 3 to 16 randomly placed instrument requests for observations and calibrations, and between 50 and 175 communications satellite passes.

The score for EO-1 includes preferences for more calibrations and observations, earlier start times for the observations, fewer solar array and aperture manipulations, lower maximum value over the entire horizon for the solar array, and higher levels of propellant.

Applying the quantile-quantile (Q-Q) test to the EO-1 hypotheses shows that they are very likely normal distributions. The Q-Q test compares the quantiles of the samples with a normal distribution, and departures in linearity of the resulting plot show how the samples differ from a normal distribution. Results of applying the Q-Q test to these two domains are shown in (Gratch & DeJong 1994).

Figures 4 and 5 show scores of the generated heuristic combinations over 35 cycles of the search algorithms. Although the curves for the scores of the two different search algorithms are different, the percentage of improvement for the high scoring hypothesis within each cycle is similar (128% for the linear search compared with 147% for the genetic algorithm). The percentage improvement for the mean score is somewhat greater, 161% for the genetic algorithm compared with 116% for the linear search. The

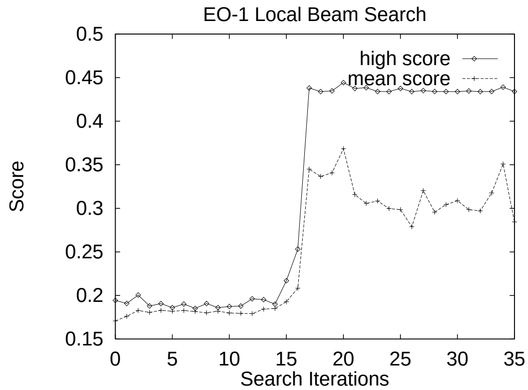

Figure 4: EO-1 model search iteration maximum and average scores for 35 iterations of the local beam search (beam = 2).

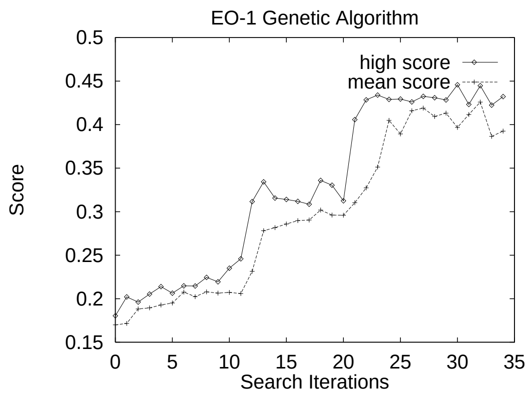

Figure 5: EO-1 model search iteration maximum and average scores for 35 iterations of the genetic algorithm search.

high scoring heuristic combinations are also somewhat different: the local search hypotheses use a significantly lower percentage of random heuristics than the genetic algorithm hypotheses, illustrating two different local maxima in the search space.

New Millennium Space Technologies Four Landed Operations Domain– The ST-4 domain models the landed operations of a spacecraft designed to land on a comet and return a sample to earth. This model has 6 shared resources, 6 state variables, and 22 activity types. Resources and states include battery level, bus power, communications, orbiter-in-view, drill location, drill state, oven states for a primary and backup oven state, camera state, and RAM. There are two activity groups that correspond to different types of experiments: mining and analyzing a sample, and taking a picture. Each ST-4 problem instance includes a randomly generated, fixed profile that represents communications visibility to the orbiting spacecraft. Each problem also includes between 1 and 11 mining activities and between 1 and 24 picture experiments at random start times.

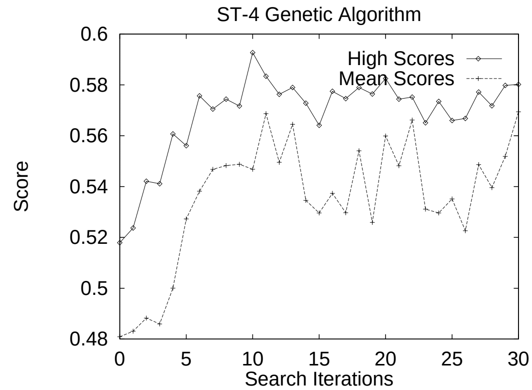

Figure 6: ST-4 model current iteration maximum and average scores for 30 genetic algorithm generations.

The preferences for ST-4 include more imaging, more mining, higher average battery power, fewer drill movements, and fewer uplinks.

Based on the Q-Q test, hypotheses from the ST-4 domain are likely to be normally distributed, and thus provides a good model for adaptive problem solving (Gratch & DeJong 1994). Graph 6 shows the mean and high scores of the generated heuristic combinations over 25 cycles of the search algorithms. The high score reaches a maximum improvement of 14%, and the mean score has a maximum improvement of 18%.

Related Work

Evaluating control strategies is a growing research topic. Horvitz originally described a method for evaluating algorithms based on a cost versus quality tradeoff (Horvitz 1988). Russell, Subramanian, and Parr used dynamic programming to rationally select among a set of control strategies by estimating utility (which includes cost) (Russell *et al.* 1993). The MULTI-TAC system considers all k-wise combinations of heuristics for solving a CSP in its evaluation which also avoids problems with local maxima, but at a large expense to the search (Minton 1996). Fink describes a method that sets time bounds for selection as opposed to parameter estimation accuracy, since sampling time is not large enough to attempt to minimize the number of samples (Fink 1998). Previous articles describing adaptive problem solving have developed general methods have been developed for transforming a standard problem solver into an adaptive one(Gratch & DeJong 1992; 1996), illustrated the application of adaptive problem solving to real world scheduling problems (Gratch & DeJong 1996), and showed how adaptive problem solving can be cast as a resource allocation problem (Gratch & DeJong 1994). We expand on these topics by evaluating different methods for generating hypotheses which can be used in adaptive problem solving to efficiently estimate their utility and cost, considered separately.

Future Work

In the area of adaptive problem solving, additional work has been proposed for the stopping criteria, which can be resource bounded (specifically, time as a resource) instead of a relaxation of the ranking requirement, as in previous works on similar topics (Fink 1998). Different methods of combining heuristics could be applied to problems of this type. One method is composite strategies, from operations research, which involve logical decisions about the relative usage of heuristics as opposed to statistical methods. Another method is a portfolio approach, which combines heuristics in a method similar to a financial portfolio.

Current results do not indicate any direct benefit to using either local beam search or genetic algorithms over the alternative. In order to predict an effective search algorithm for each environment, it would be useful to generate a landscape of the utilities for the hypothesis space (Wolpert 1996), and choose the appropriate search algorithm for the environment. Previous work has been done in deterministic landscape generation (Wolpert 1996; Whitley 1995), but no practical work has been done in stochastic landscape generation, which is what this domain requires.

Conclusions

This paper outlines different methods for adapting control strategies using adaptive problem solving, with the goal of finding a control strategy or set of control strategies that performs well in the given planning and scheduling environment. The purpose is validated in all three planning and scheduling domains, by showing significant overall improvement in the generated plans. These results are significant in showing that autonomous spacecraft planning and scheduling is becoming a realistic option for missions to unknown environments.

ACKNOWLEDGMENTS

This work was performed by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

References

Chien, S.; Gratch, J.; Burl, M. 1995. On the Efficient Allocation of Resources for Hypothesis Evaluation: A Statistical Approach. In *Proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence 17(7)*. 652–665

Fink, E. 1998. How to Solve it Automatically: Selection among Problem-Solving Methods. In *Proceedings of the Fifth International Conference of AI Planning Systems*, 128–136.

Fukunaga, A.; Rabideau, G.; Chien, S.; and Yan, D. 1997. Towards an application framework for automated planning and scheduling. In *Proceedings of the 1997 International Symposium on Artficial Intelligence, Robotics and Automation for Space*.

Goldberg, D. 1989. *Genetic Algorithms: In Search, Optimization and Machine Learning*. Reading, Massachusetts: Addison–Wesley.

Gratch, J.; DeJong, G. 1996. A Decision-theoretic Approach to Solving the EBL Utility Problem. In *Artificial Intelligence* .

Gratch, J.; Chien, S.; DeJong, G. 1994. Improving Learning Performance Through Rational Resource Allocation. In *Proceedings of the Twelfth National Conference on Artificial Intelligence* , pp. 576-582.

Horvitz, E. 1988. Reasoning under Varying and Uncertain Resource Constraints. *Proceedings of the Seventh National Conference on Artificial Intelligence* , 111–116.

Minton, S. 1996. Automatically Configuring Constraint Satisfaction Programs: A Case Study. In *Constraints* 1:1(7-43).

Minton, S. 1988. Minimizing Conflicts: A Heuristic Repair Method for Constraint Satisfaction and Scheduling Problems. *Artificial Intelligence* 58:161–205.

Rabideau, G.; Chien, S.; Backes, G.; Chalfant, G.; Tso, K. 1999. A Step Towards an Autonomous Planetary Rover. *Space Technology and Applications International Forum* .

Russell, S.; Norvig, P. 1995. *Artificial Intelligence: A Modern Approach*. Upper Saddle River, NJ: Prentice Hall.

Russell, S.; Subramanian, D.; and Parr, R. 1993. Provably Bounded Optimal Agents. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*.

Sherwood, R.; Govindjee, A; Yan, D.; Rabideau, G.; Chien, S.; Fukinaga, A. 1998. Using ASPEN to automate EO-1 Activity Planning. In *Proceedings of the 1998 IEEE Aerospace Conference*.

Smith, S. 1994. OPIS: A Methodology and Architecture for Reactive Scheduling. In Zweben, M. and Fox, M., eds., *Intelligent Scheduling* . San Francisco: Morgan Kaufmann. 29-66.

Whitley, D.; Mathias, K.; Rana, S.l Dsubera, J. 1995. Building Better Test Functions In *Procedings of the International Conference on Genetic Algorithms*.

Wolpert, D.H.; Macready W.G. 1996. No Free Lunch Theorems for Search *SFI-TR-95-02-010*.

JPL 1999. *http://www.jpl.nasa.gov/missions/*.

Zweben, M.; Daun, B.; Davis, E.; and Deale, M. 1994. Scheduling and rescheduling with iterative repair. In Zweben, M., and Fox, M., eds., *Intelligent Scheduling*. San Francisco, CA: Morgan Kaufmann. 241–256.